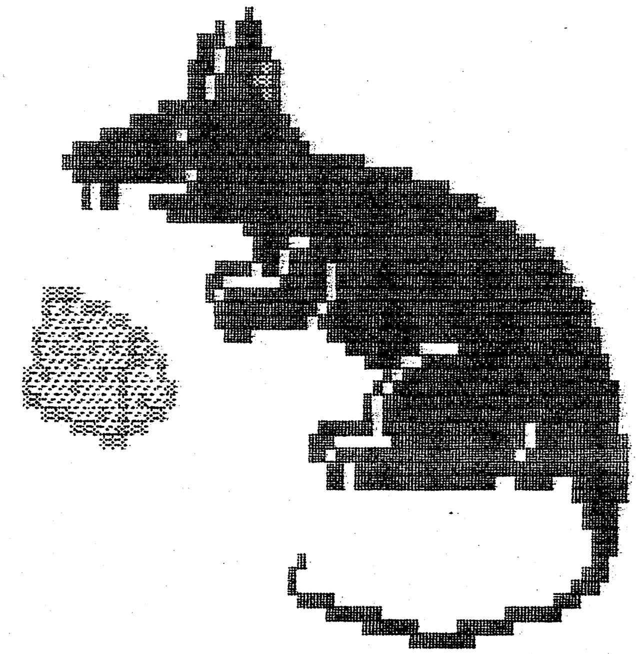
JUNE 1992 ISSUE # 39

VZ 200 ♦ VZ 300

HUNTER VALLEY

VZ JOURNAL

VZ



HAPPY 6TH BIRTHDAY

PRODUCED BI-MONTHLY BY H.V.VZ.U.G.

A NON PROFIT ORGANIZATION

FRONT COVER

WE HAVE JASON OAKLEY'S ARTISTIC TALENTS TO THANK FOR THE NEW FRONT COVER. IN CASE YOU HAVEN'T WORKED IT OUT THE RODENT SIGNIFIES A VZ MOUSE.

HELP - SELL & TELL — PAGE 3

SIXTH BIRTHDAY, LAST MEETING, WA VISITOR, MORE PUBLIC DOMAIN, WANTED DISK DRIVE SYSTEM AND SOURCE CODE ROUTINES.

VZ MOUSE BY GARY BULLEY — PAGES 4-9

AT LONG LAST AFTER MANY REQUESTS THE MOUSE PROJECT IS A REALITY. MAKE SURE YOU READ EDITOR'S WARNING AND NOTE ON 2ND MOUSE PROJECT ON PAGE 8 BEFORE PROCEEDING WITH CONSTRUCTION.

KSCAN PT II BY LESLIE MILBURN — PAGES 11-12

LESLIE CONTINUES HIS ARTICLE ON ENHANCING VZ KEYBOARD INPUT WHICH IS ALSO CONTINUED IN NEXT ISSUE. IF LESLIE'S PLANS COME TO FRUITION THEN THE HUMBLE VZ WILL COME CLOSE TO THAT OF AN IBM TYPE COMPUTER. I FOR ONE WISH HIM LUCK.

ASSEMBLY LANGUAGE PART II BY BOB KITCH — PAGES 13-16

PART II OF BOB'S ASSEMBLY TUTORIAL TAKES US FURTHER ALONG THE ROAD IN UNDERSTANDING ASSEMBLER PROGRAMMING. WELL WORTH THE READ.

QUICKWRITE INFORMATION BY LESLIE MILBURN — PAGES 17-18

LESLIE PROVIDES US WITH INFORMATION ON QUICKWRITE WORD PROCESSOR VERSIONS AND THEIR STRUCTURE.

DAVE MITCHELL SOFTWARE FOR SALE — PAGE 19
PATCH3.3 - EXT DOS & MENU/FILE COPIER

PETER HICKMAN SOFTWARE FOR SALE — PAGE 19
VZ MODEM & M/C DISASSEMLER

USER GROUPS ** NEWS ** SUBSCRIPTIONS — PAGE 20

BELIEVE IT OR NOT:

A WEEK AGO I TOOK MY CAR FOR A REGO CHECK AND AS A RESULT BOUGHT 5 NEW TYRES AS THE OLD ONES WOULD NOT HAVE LASTED MORE THAN SIX MONTHS. NEXT STOP WAS TO NRMA TO GET MY GREEN SLIP. THEY COULDN'T FIND NO RECORD OF MY PREVIOUS ONE SO ASKED TO SEE MY OLD REGO PAPERS.

IT DIDN'T TAKE LONG FOR THEM TO FIND OUT THAT MY REGO WAS STILL 6 MONTHS AWAY AND THAT I HAD MY SON'S REGO PAPERS. THE LESSON I LEARNED FROM THE MIXUP IS THAT IN FUTURE I MUST PUT ON MY READING GLASSES AND GET STUCK INTO SILICON TO IMPROVE MY MEMORY CHIPS.

CLUB & JOURNAL 6TH BIRTHDAY:

AS THE FRONT COVER SHOWS THE HUNTER VALLEY VZ JOURNAL IS SIX YEARS OLD AND IS THE LONGEST RUNNING VZ PUBLICATION TO DATE. IT HAS BEEN A VERY ENJOYABLE 5 1/2 YEARS FOR ME AS EDITOR AND EVEN AFTER ALL THIS TIME I'M STILL LEARNING ABOUT THE VZ AND INTEND TO KEEP GOING FOR SOME TIME YET.

MY SINCERE THANKS TO OUR PAST AND PRESENT CONTRIBUTORS TO THE JOURNAL FOR THEIR SUPPORT VIA THEIR EXELLENT ARTICLES, ASSISTANCE TO MYSELF WHICH HAS MADE MY JOB SO MUCH EASIER. IN NO PARTICULAR ORDER I THANK THE FOLLOWING AND MY APOLOGIES IF I LEFT SOMEONE OUT.

DAVE MITCHELL, BRIAN GREEVE, ROBERT QUINN, BOB KITCH, GARY BULLEY, DAVE BOYCE, LESLIE MILBURN, JASON OAKLEY, LARRY TAYLOR, PETER HICKMAN, RUSSELL HARRISON AND NEVILLE HUGHES. THANK YOU FELLAS.

A SPECIAL THANKS GOES ALSO TO OUR LOCAL, INTRASTATE, INTERSTATE AND INTERNATIONAL MEMBERS FOR WITHOUT WHOSE CONTINUED SUPPORT THERE WOULD BE NO JOURNAL OR CLUB.

LAST MEETING:

OUR LAST MEETING WAS HELD AT ROSS WOOD'S PLACE AND MOST OF THE EVENING WAS SPENT WITH AN IBM COMPATIBLE COMPUTER WHICH THE MAJORITY OF OUR LOCAL MEMBERS HAVE. OUR VZ GROUP WAS ORIGINALLY FORMED SO WE COULD LEARN MORE ABOUT VZ COMPUTERS. WE STILL ARE A VZ CUMPUTER CLUB WITH THE SAME AIM AS BEFORE, EXCEPT THAT AS WE LEARN WE NOW USE AND EN-COMPASS OTHER COMPUTERS AS WELL.

WA VISITOR:

BRIAN GREEVE AND HIS FAMILY FROM WESTERN AUSTRALIA CALLED IN COUPLE MONTHS AGO TO SAY HELLO. UNFORTUNATELY TIME DID NOT PERMIT THEM TO STAY MORE THAN A COUPLE HOURS WHICH WENT ALL TO QUICKLY. IT IS ALWAYS A PLEASURE TO MEET FELLOW VZ USERS. AS MENTIONED BEFORE IF YOU ARE IN THE NEIGHBOURHOOD PLEASE DO CALL IN TO SAY HELLO AND STAY TO HAVE A CUPPA AT LEAST.

MORE PUBLIC DOMAIN:

A COUPLE MORE VZ SOFTWARE AUTHORS HAVE DECLARED THEIR PROGRAMS PUBLIC DOMAIN. THE PROGRAMS IN QUESTION ARE TWO OF THE MOST SOPHISTICATED EVER WRITTEN FOR THE VZ COMPUTERS. THEY ARE:

1) ROBERT QUINN DATABASE (TAPE/DISK VERSIONS)

2) LES MILBURN QUICKWRITE WORD PROCESSOR SERIES (DETAILS PAGES 17-18)

FOR MORE INFORMATION CONTACT EDITOR.

WANTED DISK DRIVE SYSTEM:

IF YOU HAVE A SYSTEM FOR SALE THEN PLEASE LET ME KNOW AS I HAVE HAD SEVERAL ENQUIRIES FROM MEMBERS WISHING TO PURCHASE A SYSTEM. EDITOR.

SOURCE CODE ROUTINES:

THE RESPONSE TO LAST ISSUE'S REQUEST WAS DISSAPOINTING WITH ONLY TWO MEMBERS PROMISING SOME ROUTINES. THE IDEA WAS AND IS FOR ALL VZ USERS TO SHARE AND LEARN FROM EACH OTHER'S ROUTINES. THE REQUEST STILL STANDS. IF YOU CAN HELP OUT PLEASE CONTACT EDITOR.

VZ 200/300 MOUSE BY GARY BULLEY 39/4

SOME TIME AGO I STARTED ON A PROJECT TO INSTALL A MOUSE ON THE VZ. THIS LED TO A SERIES OF PROJECTS WHICH DELAYED THE END RESULT SOMEWHAT BUT OVER THE NEXT FEW EPISODES OF THE JOURNAL I WILL DESCRIBE ALL THE PROGRESS MADE SO FAR. IT WILL BE ENOUGH TO GET THE MOUSE UP AND RUNNING AND ENABLE YOU TO DEVELOP YOUR OWN PROGRAMS FOR USING IT.

TWO MAJOR PROJECTS HAVE TO BE UNDERTAKEN. ONE IS THE 8 BIT INPUT PORT WHICH HAS ALREADY BEEN DESCRIBED IN A PREVIOUS ISSUE. THE OTHER IS MODIFICATIONS TO THE MOUSE.

THE MOUSE I USED WAS PURCHASED FROM TANDY AND IS A SERIAL MOUSE CAT NO. 25-1040C 10A9. BECAUSE THE MOUSE HAS TO BE MODIFIED I THINK ANY TYPE OF CHEAP MOUSE ON TO-DAY'S MARKET WOULD BE SUITABLE. THE MODIFIED CIRCUIT IS MADE UP ON A SMALL PIECE OF VERO BOARD AND HOUSED INSIDE THE MOUSE CASE. IT CONNECTS BY FLYING LEADS TO THE APPROPRIATE POINTS ON THE MOUSE HARDWARE.

THE FOLLOWING IS A DESCRIPTION OF HOW THE MOUSE WORKS:

INSIDE THE MOUSE THERE ARE TWO CIRCULAR DISKS. ONE DISK FOR THE HORIZONTAL TRAVEL AND THE OTHER FOR VERTICAL. AROUND THE CIRCUMFERENCE OF THESE DISKS ARE CUT EVENLY SPACED SLOTS. THE DISK IS ARRANGED SO THAT AS IT ROTATES THE SLOTS WILL INTERRUPT A LIGHT BEAM THAT IS FOCUSSED ON A PAIR OF LIGHT SENSING TRANSISTORS.

EACH DISK IS DRIVEN BY A SHAFT WHICH IS IN CONTACT WITH A RUBBER BALL. NOW AS THE MOUSE IS MOVED THE BALL WILL ROTATE WHICH IN TURN ROTATES THE DISK AND THE RESULT IS A SQUARE WAVE OUTPUT FROM THE TRANSISTORS THAT IS PROPORTIONAL TO MOUSE MOVEMENT. THIS SQUARE WAVE CAN NOW BE USED TO TRIGGER A LATCH WHICH WILL INDICATE MOUSE MOVEMENT AND DIRECTION.

AS THE DISK IS ROTATED IN ONE DIRECTION THE BEAM WILL BE CUT WHICH WILL BLOCK THE TRANSISTORS FROM ANY LIGHT AND THEIR OUTPUT WILL GO TO A LOGIC ONE. FURTHER ROTATION WILL ALLOW LIGHT TO PASS THROUGH A SLOT AND EXPOSE THE TRANSISTOR TO THE BEAM AND NOW IT'S OUTPUT WILL DROP TO A LOGIC ZERO. THIS IS HOW THE SQUARE WAVE IS PRODUCED AND WILL CONTINUE AS LONG AS THE DISK IS ROTATED. THE SAME WILL APPLY WHEN THE DISK IS ROTATED IN THE OPPOSITE DIRECTION, A SQUARE WAVE OUTPUT WILL STILL BE PRODUCED.

NOW IF THE OUTPUT OF A LATCH IS SET TO A ZERO THIS SQUARE WAVE THAT IS PRODUCED BY THE MOUSE CAN TRIGGER THE LATCH AND SET IT'S OUTPUT TO A ONE, SO BY MONITORING THE OUTPUT OF THE LATCH WE CAN TELL WHEN THE DISK HAS ROTATED. I.E. IF THE LATCH OUTPUT REMAINS AT ZERO THEN NO MOVEMENT HAS OCCURRED.

THIS LEAVES THE DIRECTION OF MOVEMENT TO BE DETERMINED. TO DO THIS A SECOND LATCH IS USED ALONG WITH THE SECOND TRANSISTOR BUT THIS TIME THE TRANSISTOR IS USED TO SET THE OUTPUT OF THE LATCH AT EITHER A ONE OR ZERO DEPENDING ON WHETHER THE TRANSISTOR IS EXPOSED TO THE LIGHT OR CUT OFF.

THE TWO TRANSISTORS ARE POSITIONED SIDE BY SIDE EACH OTHER IN THE PATH OF THE BEAM. AS THE DISK IS ROTATED IN ONE DIRECTION A STAGE IS REACHED WHERE BOTH TRANSISTORS ARE EXPOSED TO THE BEAM. FURTHER ROTATION WILL CUT OFF ONE TRANSISTOR (CALL IT TRANSISTOR A) AND IT'S OUTPUT WILL RISE TO A ONE. AS A IS USED TO TRIGGER THE MOVEMENT LATCH THE OUTPUT OF THE LATCH WILL RISE TO A ONE AND THIS RISING OUTPUT CAN THEN BE USED TO TRIGGER THE SECOND (DIRECTION) LATCH.

BECAUSE THE SECOND TRANSISTOR (B) IS STILL EXPOSED TO THE BEAM, IT'S OUTPUT WILL BE ZERO AND SO THE OUTPUT OF THE DIRECTION LATCH IS SET AT ZERO. FURTHER ROTATION WILL EVENTUALLY CUT THE BEAM TO B BUT THE OUTPUT STATE OF THE LATCHES WILL NOT CHANGE AND WILL STAY LIKE THIS WHILE EVER THE DISK ROTATES IN THIS DIRECTION. WHEN THE DISK IS ROTATED IN THE OPPOSITE DIRECTION TRANSISTOR A WILL STILL TRIGGER THE MOVEMENT LATCH AND B THE DIRECTION LATCH, BUT THIS TIME TRANSISTOR B WILL BE CUT FROM THE BEAM BEFORE A SO IT'S OUTPUT WILL BE A ONE WHICH WILL SET THE DIRECTION LATCH TO A ONE WHEN IT IS TRIGGERED.

IN THE CIRCUT DIAGRAM FOR THE MOUSE THE OUTPUT FROM THE FOUR LATCHES IS LOADED IN PARALLEL ALONG WITH THE TWO PUSH BUTTONS INTO THE 74LS165 CHIP AND IS THEN CLOCKED OUT SERIALLY TO THE INPUT PORT OF THE VZ. ONCE INSIDE THE VZ IT IS TREATED BY SOFT WARE AS AN 8 BIT WORD AND THE BITS CHECKED INDIVIDUALLY FOR MOUSE MOVEMENT AND DIRECTION AS WELL AS PUSH BUTTON OPERATION.

CIRCUIT BOARD:

STUDY THE CIRCUIT BOARD (VERO BOARD) CAREFULLY UNTIL YOU ARE SURE OF ALL THE CONNECTIONS. THERE ARE A NUMBER OF BRIDGES TO GO IN BEFORE THE IC'S ARE SOLDERED IN PLACE. BECAUSE SIZE AND SPACE ARE AT A PREMIUM THE BOARD IS A BIT CRAMPED. USE SMALL GAUGE LEADS AND A GOOD CLEAN SOLDERING IRON FOR THE BEST RESULTS. THE CONNECTION BETWEEN BRIDGE B AND PIN 3 OF THE 74LS165 IS A BIT TRICKY.

PASS THE WIRE THROUGH THE HOLE NEAR PIN 3 THEN BEND IT OVER ON THE COPPER SIDE OF THE BOARD AND SOLDER IT TO PIN 3. YOU WILL HAVE TO WORK OUT THE TERMINATION POINTS FOR THE HORIZONTAL, VERTICAL AND PUSH BUTTONS ON THE MOUSE THAT YOU PURCHASE. IF YOU HAVE A CIRCUIT DIAGRAM IT WILL BE A BIG HELP OTHERWISE A GOOD MAGNIFYING GLASS, PENCIL AND PAPER WILL BE THE ONLY ALTERNATIVE.

8 BIT INPUT PORT MODS:

THERE IS ONE MODIFICATION TO BE MADE TO THE 8 BIT INPUT PORT. THE STROBE SIGNAL (PIN 12/74LS138) HAS TO BE CONNECTED TO PIN 11 OF DB15. REFER TO ISSUE 30, PAGES 9-11 FOR MORE DETAILS ON 8 BIT INPUT PORT.

VZ MOUSE SOURCE CODE:

```
;***VZ MOUSE***
; ORIGIN 6000H
; WRITTEN  BY
; GARY BULLEY
; (AUTO START ON RESET)
     DEFB 0AAH        ; FIRST FOUR IDENTITY BYTES
     DEFB 55H           FOR ROM IPL SEQUENCE.
     DEFB 0E7H
     DEFB 18H
     LD   HL,STRT     ; LOAD INTERUPT VECTOR
     LD   (787EH),HL    WITH MOUSE START ADDRESS.
     LD   A,0C3H
     LD   (787DH),A
     JP   1A19H       ; JUMP TO READY.
;MOUSE INPUT
STRT LD   HL,INPT     ; LOAD 8 BIT WORD FROM
     LD   B,08H         MOUSE AND STORE AT
     OUT  (1),A         INPUT LOCATION.
LOOP IN   A,(1)
     RRA
```

```
        RR    C
        DJNZ  LOOP
        LD    (HL),C          ; REG C=INPUT WORD.
        BIT   5,C             ; TEST BUTTON 2.
        JR    NZ,CRSR         ; JUMP IF NOT PUSHED.
        LD    HL,FLAG         ; TEST FLAG FOR
        BIT   0,(HL)            SPECIAL FUNCTION MODE.
        RET   NZ              ; RETURN IF YES.
        POP   HL              ; STACK THE STACK
        POP   DE                WITH SPECIAL FUNCTION
        POP   BC                ADDRESS.
        POP   AF
        EXX
        POP   DE
        LD    HL,SPFN
        PUSH  HL
        PUSH  DE
        EXX
        PUSH  AF
        PUSH  BC
        PUSH  DE
        PUSH  HL
        RET                   ; RETURN.
;MOVE VERTICAL
CRSR    INC   HL              ; HL=VERTICAL COUNTER.
        BIT   1H,C            ; TEST VERTICAL MOVEMENT.
        JR    Z,HORZ          ; JUMP IF NONE.
        DEC   (HL)            ; DECREMENT VERTICAL COUNT.
        RET   NZ              ; RETURN IF NOT ZERO.
        INC   (HL)            ; INCREMENT VERTICAL COUNT.
        INC   HL              ; HL=HORIZONTAL COUNTER.
        LD    (HL),3H         ; LOAD COUNTER TO 3.
        BIT   0H,C            ; TEST VERTICAL DIRECTION.
        LD    A,(783CH)       ; REPLACE OLD CURSOR
        LD    HL,(7820H)        CHARACTER.
        LD    (HL),A
        LD    BC,0020H        ; BC=ONE SCREEN LINE.
        JR    NZ,DOWN         ; JUMP IF DIRECTION IS DOWN.
        XOR   A               ; CALCULATE NEW
        SBC   HL,BC             CURSOR POSITION.
        LD    A,H
        CP    70H             ; RETURN IF ABOVE
        RET   M                 TOP OF SCREEN.
        JR    CHAR            ; JUMP TO PRINT CHARACTER.
DOWN    XOR   A               ; CALCULATE NEW
        ADC   HL,BC             CURSOR POSITION.
        LD    A,H
        CP    72H             ; RETURN IF BELOW
        RET   P                 BOTTOM OF SCREEN.
        JR    CHAR            ; JUMP TO PRINT CHARACTER.
;HORIZONTAL
HORZ    BIT   3H,C            ; TEST HORIZONTAL MOVEMENT.
        JR    NZ,MOVE         ; JUMP IF YES.
        LD    A,01            ; LOAD VERTICAL
        LD    (HL),A            COUNT TO 1.
        INC   HL              ; HL=HORIZONTAL COUNTER.
        LD    (HL),A          ; LOAD COUNT TO 1.
        RET                   ; RETURN.
MOVE    INC   HL              ; HL=HORIZONTAL COUNT.
        DEC   (HL)            ; DECREMENT HORIZONTAL COUNT.
        RET   NZ              ; RETURN IF NOT ZERO.
        INC   (HL)            ; INCREMENT HORIZONTAL COUNT.
```

```
        DEC  HL           ; HL=VERTICAL COUNTER.
        LD   (HL),03H     ; LOAD COUNTER TO 3.
        LD   A,(783CH)    ; REPLACE OLD CURSOR
        LD   HL,(7820H)     CHARACTER.
        LD   (HL),A
        LD   A,1FH        ; CALCULATE CURSOR
        AND  L              POSITION.
        BIT  2H,C         ; TEST HORIZONTAL POSITION.
        JR   NZ,RGHT      ; JUMP IF RIGHT.
        DEC  A            ; CALCULATE NEW POSITION.
        RET  M            ; JUMP IF LEFT OF SCREEN.
        CALL 3227H        ; CALL ROM MOVE LEFT ROUTINE.
        JR   FINS         ; JUMP TO FINISH.
   RGHT XOR  1FH          ; CALCULATE NEW POSITION.
        RET  Z            ; JUMP IF RIGHT OF SCREEN.
        CALL 31B8H        ; CALL ROM MOVE RIGHT ROUTINE
        JR   FINS         ; JUMP TO FINISH.
   CHAR LD   A,(HL)       ; STORE NEW CURSOR
        LD   (783CH),A      CHARACTER AND
        LD   (7820H),HL     POSITION.
   FINS LD   A,40H        ; FLASH NEW CURSOR CHARACTER.
        XOR  (HL)
        LD   (HL),A
        LD   A,10H        ; RESET CURSOR FLASH RATE.
        LD   (7841H),A
        RET               ; RETURN.
   FLAG NOP               ; FLAG WORD.
   INPT DEFB 00           ; INPUT WORD LOCATION.
   ;VERTICAL COUNT
        DEFB 01H          ; VERTICAL COUNTER.
   ;HORIZONTAL COUNT
        DEFB 01H          ; HORIZONTAL COUNTER.
   ;SPEC FUNCTION
   SPFN PUSH AF           ; START OF SPECIAL
        PUSH BC             FUNCTION ROUTINE.
        PUSH DE
        PUSH HL
        LD   HL,FLAG
        SET  0,(HL)
        LD   A,90H
        XOR  (IY+0BH)
        LD   (IY+0BH),A
        INC  HL
   KPSH BIT  5,(HL)
        JR   Z,KPSH
        DEC  HL
        RES  0,(HL)
        POP  HL
        POP  DE
        POP  BC
        POP  AF
        RET
```

PROGRAM NOTES:

THE PROGRAM HAS BEEN WRITTEN TO RESIDE AT 6000H BUT BECAUSE THERE IS NO MEMORY AT THIS LOCATION IN A STANDARD VZ IT WILL REQUIRE A MODIFICATION TO BE DONE IF YOU WANT TO USE THIS LOCATION. THE PROGRAM COULD BE RELOCATED ANYWHERE IN MEMORY IF YOU SO DESIRED. THE REASON I CHOSE THIS LOCATION IS BECAUSE VERY FEW PROGRAMS USE THIS SECTION OF MEMORY SO THE PROBLEMS OF IT BEING CORRUPTED OR CORRUPTING OTHER PROGRAMS IS VIRTUALLY NILL.

THE OTHER REASON IS THAT THE IPL SEQUENCE CHECKS FOR A PROGRAM AT 6000H AND WILL AUTOMATICALLY RUN THE PROGRAM IF THE FIRST 4 IDENTITY BYTES ARE CORRECT. THIS WAY EVERY TIME THE RESET BUTTON IS PUSHED THE MOUSE PROGRAM WILL BE UP AND RUNNING AND WILL NOT HAVE TO BE RELOADED EACH TIME.

THE PROGRAM AND DESCRIPTION IS REASONABLY STRAIGHT FORWARD BUT THERE ARE A FEW THINGS THAT MAY NEED EXPLAINING. IF YOU ARE GOING TO RE-ASSEMBLE IT FOR ANOTHER SECTION OF MEMORY THEN THE PROGRAM WILL START AT LINE 10. THE 8 BIT WORD THAT IS INPUT FROM THE MOUSE CONTAINS ALL THE INFORMATION FOR CURSOR MOVEMENT AND PUSHBUTTON OPERATION.

I HAVE WRITTEN A SMALL PROGRAM THAT I HAVE CALLED "SPECIAL FUNCTION MODE" AND USES BUTTON 2 TO CHANGE DISK DRIVES. IT IS AN EXAMPLE OF HOW THE BUTTONS CAN BE USED AND USES A PROGRAM CALLED "STACK THE STACK". IT WORKS LIKE THIS EVERY TIME A SOFT WARE INTERRUPT OCCURRS THE MOUSE ROUTINE IS RUN AND THE BUTTONS SCANNED.

IF BUTTON 2 IS PUSHED THE STACK IS LOADED WITH THE ADDRESS OF THE SPECIAL FUNCTION PROGRAM WHICH WILL CAUSE THE PROGRAM TO RETURN BACK TO IT INSTEAD OF THE MAIN PROGRAM. ONCE THE SPECIAL FUNCTION PROGRAM IS FINISHED THE PROGRAM WILL RETURN TO ITS CORRECT ADDRESS. SEE BOB KITCH'S DESCRIPTION IN ISSUE 24 "VECTORS AND INTERRUPTS EXPLAINED".

THE REST OF THE PROGRAM LOOKS AFTER THE CURSOR MOVEMENT. IN ORDER TO SLOW DOWN MOUSE MOVEMENT TWO COUNTERS ARE USED. WHAT HAPPENS IS, WHEN THE MOUSE IS MOTIONLESS BOTH COUNTERS ARE LOADED WITH A COUNT OF 1. WHEN MOVEMENT IS DETECTED IN ANY DIRECTION THAT DIRECTION COUNTER IS STILL LOADED WITH A COUNT OF 1 AND THE OTHER DIRECTION COUNTER IS LOADED WITH A COUNT OF 3. BY DOING THIS IT LOCKS IN TO THE DIRECTION WITH THE LOWER COUNT AND BECOMES MORE STABLE IN MOVEMENT.

THE WHOLE OF THE MOUSE PROGRAM IS MACHINE CODE BUT THERE ARE FOUR LOCATIONS THAT CAN BE USED BY BASIC WITH THE PEEK COMMAND. LINE 109 IS THE FLAG LOCATION FOR THE SPECIAL FUNCTION ROUTINE. BIT 0 IS THE ONLY BIT USED SO THERE IS PLENTY OF ROOM FOR EXPANSION.

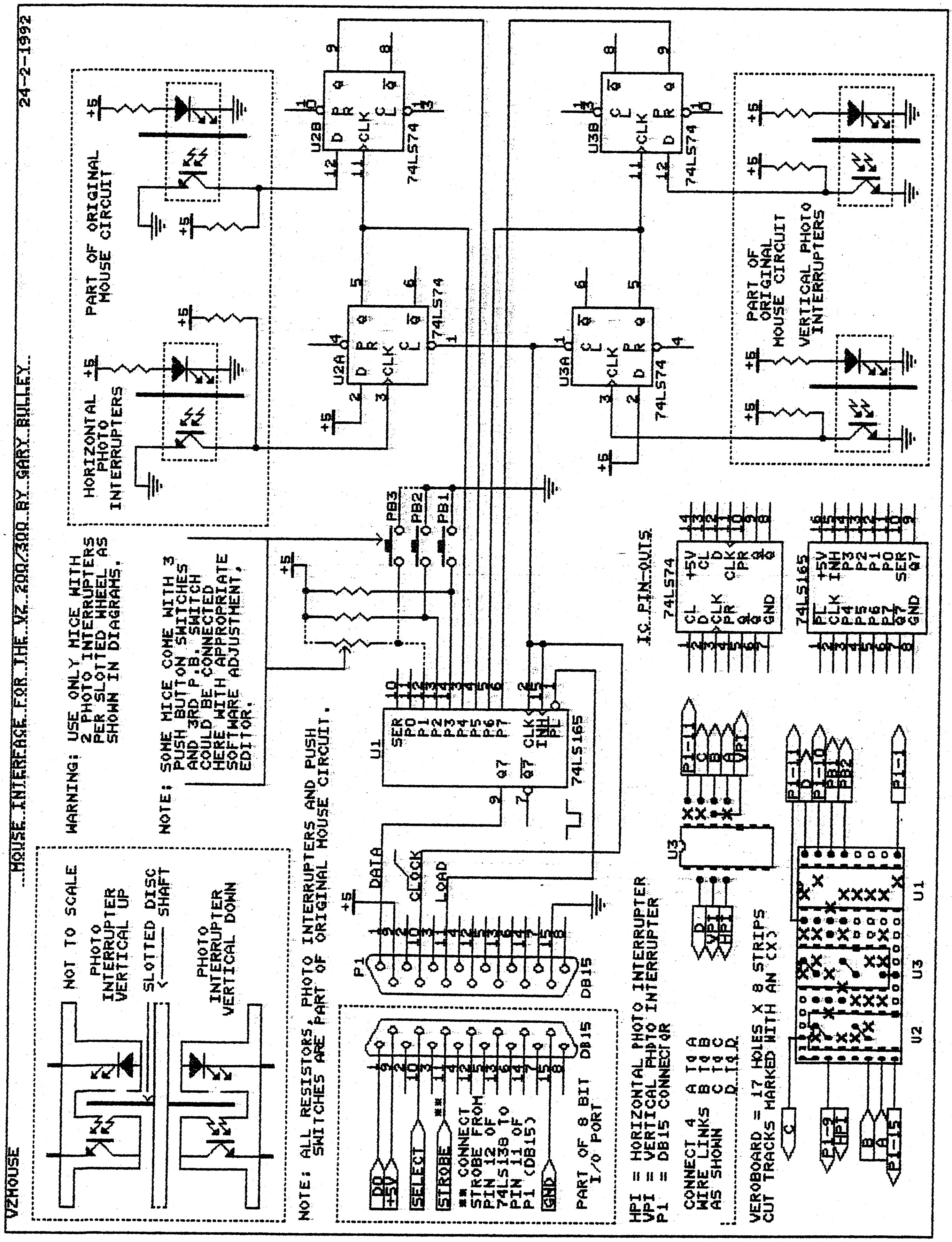
LINE 110 IS THE LOCATION OF THE 8 BIT INPUT WORD FROM THE MOUSE. THIS IS PROBABLY THE MOST IMPORTANT LOCATION AS FAR AS PROGRAMMING GOES AS BITS 0 TO 5 ARE THE MAIN INDICATORS OF WHAT THE MOUSE IS DOING. LINES 112 AND 114 ARE THE DIRECTION COUNTERS.

EDITOR'S WARNING:

FOR THE ABOVE PROJECT TO WORK CORRECTLY USE A MOUSE WHICH HAS A PHOTO INTERRUPTER OR LED/PHOTO TRANSISTOR ARRANGEMENT ON EACH SIDE OF BOTH SLOTTED DISKS. THERE ARE SOME MICE AROUND WHICH HAVE ONLY A SINGLE INTERRUPTER ON EACH SLOTTED DISK AND ARE NOT SUITABLE FOR THIS PROJECT. USUALLY WHEN YOU BUY A MOUSE A SOFTWARE PROGRAM COMES WITH IT WHICH IS USED WITH THE PARTICULAR TYPE OF MOUSE PURCHASED. THIS DOES AWAY WITH MOST INCOMPATIBILITY PROBLEMS.

2ND VZ MOUSE PROJECT:

AFTER GARY BULLEY'S VZ MOUSE ARTICLE WAS EDITED I WAS NOTIFIED OF A SECOND VZ MOUSE PROJECT DONE BY ANOTHER VZ USER. IT DIFFERS FROM GARY'S IN THAT NO MODIFICATIONS TO THE MICE ARE NECESSARY. IT ALSO USES THE 8 BIT INPUT PORT, BUT WITH A MINOR MODIFICATION. THE PROJECT SHOULD APPEAR IN NEXT ISSUE SO CONSTRUCTORS CAN MAKE THEIR CHOICE.



MOUSE INTERFACE FOR THE VZ 200/300 BY GARY BULLEY.
24-2-1992
VZMOUSE
WARNING: USE ONLY MICE WITH 2 PHOTO INTERRUPTERS PER SLOTTED WHEEL AS SHOWN IN DIAGRAMS.
NOTE: SOME MICE COME WITH 3 PUSH BUTTON SWITCHES AND 3RD P.B. SWITCH COULD BE CONNECTED HERE WITH APPROPRIATE SOFTWARE ADJUSTMENT, EDITOR.
NOTE: ALL RESISTORS, PHOTO INTERRUPTERS AND PUSH SWITCHES ARE PART OF ORIGINAL MOUSE CIRCUIT.
NOT TO SCALE
PHOTO INTERRUPTER VERTICAL UP
SLOTTED DISC SHAFT
PHOTO INTERRUPTER VERTICAL DOWN
HORIZONTAL PHOTO INTERRUPTERS
PART OF ORIGINAL MOUSE CIRCUIT
PART OF ORIGINAL MOUSE CIRCUIT VERTICAL PHOTO INTERRUPTERS
U2A U2B U3A U3B 74LS74
U1 74LS165
PB1 PB2 PB3
DATA CLOCK LOAD
P1 DB15
** CONNECT STROBE FROM PIN 12 OF 74LS138 TO PIN 11 OF P1 (DB15)
PART OF 8 BIT I/O PORT
D0 +5V SELECT STROBE GND
IC PIN-OUTS
74LS74
74LS165
HPI = HORIZONTAL PHOTO INTERRUPTER
VPI = VERTICAL PHOTO INTERRUPTER
P1 = DB15 CONNECTOR
CONNECT 4 WIRE LINKS AS SHOWN
A TO A
B TO B
C TO C
D TO D
VEROBOARD = 17 HOLES X 8 STRIPS
CUT TRACKS MARKED WITH AN (X)
U1 U2 U3

USING KSCAN WITH BASIC PROGRAMS

TO EASILY USE KSCAN WITH BASIC PROGRAMS, FOUR FUNCTIONS HAVE BEEN PROVIDED. THESE ARE AS FOLLOWS:-

(1) SETDELAY(FLAG, NO)

THIS FUNCTION ALLOWS A PROGRAM TO ALTER THE LENGTH OF THREE TIME DELAY LOOPS.

FLAG CAN ONLY BE ONE OF THE FOLLOWING (NOTE THAT THIS IS QUOTED):-

G - GENERAL DELAY. THIS OCCURS EVERY INTERRUPT.

K - KEYSTROKE DELAY. THIS OCCURS EVERY TIME A NON-ZERO KEY CODE IS RETRIEVED FROM THE KEY TABLES. THIS DELAY IS IN ADDITION TO THE GENERAL DELAY.

P - PAUSE DELAY. THIS OCCURS WHEN A KEY IS PRESSED WHICH CANCELS THE PAUSE FUNCTION.

NO IS THE TIME DELAY AMOUNT.

(2) SETADDR(FLAG, NO)

THIS FUNCTION ALLOWS A PROGRAM TO CUSTOMISE KSCAN.

FLAG CAN ONLY BE ONE OF THE FOLLOWING (NOTE THAT THIS IS QUOTED):-

I - INTERRUPT ROUTINE. BECAUSE KSCAN TAKES OVER THE INTERRUPT EXIT, IT WAS NECESSARY TO PROVIDE AN ALTERNATIVE. THE MAJOR DIFFERENCE IS THAT THIS FUNCTION IS CALLED AFTER THE KEY SCAN AND DISPLAY UPDATE. NO IS THE ADDRESS OF THE ROUTINE. NOTE THAT THE INTERRUPT ROUTINE MUST HAVE BEEN SET UP OR BLOADED PRIOR TO CALLING THIS FUNCTION.

T - ALTERNATE TABLE. THIS OPTION ALLOWS AN APPLICATION TO MAKE KSCAN USE AN ALTERNATIVE SET OF KEY TABLES. I.E. AN APPLICATION MIGHT BLOAD ITS OWN TABLES FROM DISK. NO IS THE ADDRESS OF THE KEY TABLE FOR ROW 0 AND IT IS EXPECTED THAT THE KEY TABLES ARE ONE AFTER THE OTHER IN MEMORY.

U - USER FUNCTION. THIS OPTION ALLOWS AN APPLICATION TO PROVIDE ITS OWN MACHINE CODE KEY TRAP ROUTINE. NO IS THE ADDRESS OF THE USER FUNCTION.

(3) RESETADDR(FLAG)

THIS FUNCTION ALLOWS AN APPLICATION TO CANCEL AN OPTION SET BY SETADDR(). AS ABOVE, FLAG CAN ONLY BE ONE OF THE FOLLOWING (NOTE THAT THIS IS QUOTED):-

I - INTERRUPT ROUTINE. THIS CANCELS AN APPLICATION PROVIDED INTERRUPT ROUTINE.

U - THIS CANCELS AN APPLICATION PROVIDED USER FUNCTION.

T - THIS CAUSES KSCAN TO REVERT BACK TO SCANNING THE DEFAULT KEY TABLES.

(4) X = SETKEY(ROW, COLUMN, KEYCODE, STATUS)

THIS FUNCTION ALLOWS AN APPLICATION TO CHANGE A KEY CODE IN THE SET OF KEY TABLES CURRENTLY BEING USED BY KSCAN. THE PREVIOUS KEY CODE STORED IN THE VARIABLE.

THE PARAMETERS TO THIS FUNCTION ARE AS FOLLOWS:-

ROW - THIS IS THE ROW NUMBER OF THE KEY TO BE REDEFINED IN THE KEYBOARD MATRIX. THIS MUST BE IN THE RANGE 0 - 7.

COLUMN - THIS IS THE COLUMN OF THE KEY TO BE REDEFINED IN THE KEYBOARD MATRIX. THIS MUST BE IN THE RANGE 0 - 5.

KEYCODE - THIS IS THE VALUE TO BE STORED IN THE KEY TABLE. IT MUST BE IN THE RANGE 0 - 255.

STATUS - THIS INDICATES THE ALT/CTRL/SHIFT STATUS. IT MUST BE IN THE RANGE 0 - 7.
BIT 0 INDICATES THE ALT KEY STATUS.
BIT 1 INDICATES THE CTRL KEY STATUS.
BIT 2 INDICATES THE SHIFT KEY STATUS.

Examples of the new BASIC commands

SETDELAY("K",100)

SETS THE KEYSTROKE DELAY TO APPROX. 1/10 OF A SECOND.

SETADDR("I",30719)

SETS UP AN INTERRUPT ROUTINE WHICH IS LOCATED AT ADDRESS 30719 DECIMAL.

```
POKE 30719,201:POKE 30720,1
SETADDR("I",PEEK(30719)+256*PEEK(30720))
```

THIS SETS UP AN INTERRUPT ROUTINE WHICH IS LOCATED AT THE ADDRESS STORED IN LOCATIONS 30719 AND 30720. IN THIS CASE 01C9H IS CALLED UPON EACH INTERRUPT.

X = SETKEY(2,5,"Q",4)

THIS REDEFINES THE SHIFT-7 KEY COMBINATION TO DISPLAY A "Q". THE PREVIOUS KEY CODE IS STORED IN VARIABLE X.

Using KSCAN with Machine Code Programs

TO USE KSCAN WTH M/C PROGRAMS A LITTLE MORE WORK IS REQUIRED ON THE PART OF THE PROGRAMMER. HOWEVER, AS USUAL WHEN WRITING M/C PROGRAMS THERE IS MORE FLEXIBILITY.

AS KSCAN IS RELOCATABLE, DIRECT ADDRESSING MUST NOT BE USED TO CALL THE ROUTINES OR ACCESS ANY KSCAN "VARIABLES".

THE ADDRESSES OF TWO IMPORTANT LOCATIONS HAVE BEEN STORED IN THE COMMUNICATIONS REGION. THESE ARE AS FOLLOWS:-

KFUN: 31273-75: THIS CONTAINS THE INSTRUCTION JP RFUN AND HELPS INDIRECT FUNCTION CALLS.

STRT: 31276-77: THIS IS THE ADDRESS OF KSCAN. ALL FUNCTION OFFSETS ARE RELATIVE TO THIS ADDRESS.

NOTE: THESE LOCATIONS CLASH WITH THE FIND ROUTINE. THIS SHOULD NOT BE A PROBLEM IF THE FIND ROUTINE IS LOADED FIRST.

Accessing KSCAN "variables" & functions.

AS MENTIONED ABOVE, DIRECT ADDRESSING CANNOT BE USED. TO OVERCOME THIS PROBLEM ALL VARIABLES ARE AT FIXED OFFSETS FROM THE START OF KSCAN WHICH IS STORED IN STRT. BELOW IS A LIST OF VARIABLES, THEIR OFFSETS AND A BRIEF DESCRIPTION:-

VCTR: 20 : ADDRESS OF ORIGINAL RST 10H VECTOR.
KFLG: 22 : STORES STATUS OF VARIOUS SYSTEM FUNCTIONS.
UADD: 23 : FLAG INDICATING THAT AN APPLICATION HAS SET A USER FUNCTION ADDRESS.
IADD: 24 : FLAG INDICATING THAT AN APPLICATION HAS SET AN INTERRUPT ADDRESS.
GDLY: 25 : GENERAL DELAY VALUE.
KDLY: 27 : KEYSTROKE DELAY VALUE.
PDLY: 29 : PAUSE DELAY VALUE.
INT : 31 : ADDRESS OF INTERRUPT FUNCTION.
USR : 33 : ADDRESS OF USER FUNCTION.
KROW: 35 : STORES A SNAPSHOT OF KEYBOARD MATRIX.
TBL : 43 : CONTAINS POINTERS TO KEY CODE TABLES.

ALSO STORED AT FIXED LOCATIONS ARE THE OFFSETS OF THE MOST USEFUL FUNCTIONS. THIS ALLOWS FUNCTION CALLS TO BE MADE IN THE FOLLOWING WAY:-

```
EXX
LD DE,FUNCTION OFFSET
CALL KFUN.
```

IT IS UP TO THE FUNCTION BEING CALLED TO EXCHANGE THE REGISTERS BACK AGAIN. THIS METHOD IS ADVANTAGEOUS AS REGISTER VALUES ARE PRESERVED THUS ALLOWING PARAMETER PASSING TO FUNCTIONS. THE ORDER OF ALL FUNCTION AND VARIABLE OFFSETS MUST NEVER BE ALTERED. TWO AREAS HAVE BEEN SET ASIDE FOR FUTURE EXPANSION.

Scanning the keyboard.

UNFORTUNATELY THE VZ TECHNICAL MANUAL TAUGHT A LOT OF US A BAD HABITS BY INSTRUCTING US TO CALL THE DEFAULT KEY SCAN ROUTINE IN ROM (2EF4H) DIRECTLY RATHER THAN INFORMING US THAT A BETTER METHOD EXISTS.

REGARDLESS OF WHETHER KSCAN IS INSTALLED OR NOT THE CORRECT WAY TO SCAN THE KEYBOARD IS TO CALL 002BH. THIS SCANS THE KEYBOARD VIA THE D.C.B. AND ALLOWS US TO PATCH KSCAN INTO THE SYSTEM.

CONTINUED NEXT ISSUE . . .

ASSEMBLY LANGUAGE PART II
BY BOB KITCH

WITH THAT BIT OF BACKGROUND, YOU CAN NOW EXAMINE LISTING 3. THIS IS A BASIC PROGRAM WRITTEN FOR THE "VISIBLE Z80". THE BASIC PROGRAM SIMULATES WHAT AN ASSEMBLER PROGRAM COULD LOOK LIKE THAT DOES A SCREEN FILL. THE USE OF VARIABLE NAMES ARE ESPECIALLY CHOSEN TO CORRESPOND WITH THE "MYSTERIOUS" REGISTER NAMES ON THE Z80.

BY WAY OF FURTHER BACKGROUND, THE ZILOG (THE MANUFACTURERS OF THE Z80) SPECIFICATION FOR THE LDIR INSTRUCTION IS AS FOLLOWS -

A. THE HL REGISTER POINTS TO THE SOURCE OF THE LOAD.
B. THE DE REGISTER POINTS TO THE DESTINATION OF THE LOAD.
C. THE BC REGISTER CONTAINS THE COUNT OR NUMBER OF TIMES THE LOAD IS TO TAKE PLACE.
D. THE A REGISTER IS THE TRANSFER POINT BETWEEN THE SOURCE (HL) TO DESTINATION (DE) LOCATIONS.
E. AFTER EACH TRANSFER, THE HL AND DE REGISTERS ARE INCREMENTED AND THE BC (COUNTER) REGISTER IS DECREMENTED.
F. THE PROGRESSIVE LOADING IS REPEATED UNTIL THE BC REGISTER IS DECREMENTED TO ZERO - IN WHICH CASE THE INSTRUCTION IS TERMINATED.

CONDITIONS A. TO C. ARE THE INITIALIZATION FOR THE LDIR INSTRUCTION. THE ACTUAL EXECUTION OF THE INSTRUCTION INVOLVES CONDITIONS D. TO F. THE WAY IN WHICH ZILOG EXPRESS ALL OF THIS IS A LITTLE CRYPTIC BUT FOR THOSE OF YOU INTERESTED IT IS AS FOLLOWS -

(DE)<-(HL), DE<-DE+1, HL<-HL+1, BC<-BC-1, REPEAT until BC=0.

I TRUST THAT YOU CAN FIGURE SOME CORRESPONDENCE BETWEEN THIS CONCISE DESCRIPTION AND MY "WORD PICTURE" PROVIDED ABOVE.

IN LINES 200 TO 460, THE BASIC PROGRAM WITH THE CORRESPONDING PSUEDO-ASSEMBLER IS GIVEN. TRY TO FOLLOW THE RELATIONSHIP BETWEEN THE TWO LANGUAGES. READING AND UNDERSTANDING OTHER PROGRAMMERS' CODE IS A VERY USEFUL FORM OF LEARNING. I MENTIONED THAT THE ASSEMBLER LDIR INSTRUCTION IS QUITE POWERFUL, IT TAKES 4 BASIC COMMANDS TO EXECUTE IT! THIS IS UNUSUAL, AS NORMALLY ONE ASSEMBLER INSTRUCTION WILL TRANSLATE TO ONE BASIC COMMAND. A FULL DISCUSSION OF THE ASSEMBLER ROUTINE IS GIVEN IN THE NEXT SECTION.

THE ALGORITHM USED IN THIS EXAMPLE IS SIMPLE TO UNDERSTAND USING THE CHARACTERISTICS OF THE LDIR INSTRUCTION. THERE ARE MANY OTHER ROUTINES THAT WOULD ACHIEVE THE SAME JOB. IT IS IMPORTANT TO REALIZE THAT THERE IS NEVER MERELY ONE WAY TO ACHIEVE A PARTICULAR END. THE PROGRAMMER GENERALLY CALLS UPON HIS EXPERIENCE WITH FAMILIAR COMMANDS, AND THESE MAY BE THE QUICKEST AND SHORTEST WAY OF ACHIEVING THE OBJECT. BUT THERE IS NEVER ONLY ONE CORRECT SOLUTION TO CODING A PARTICULAR PROBLEM. ANOTHER ALGORITHM THAT WILL ACHIEVE A SCREEN FILL IS AS FOLLOWS-

```
   LD HL,7000H
   LD BC, 800H
LP LD (HL),170D
   INC HL
   DEC BC
   LD A,C
   OR B
   JR NZ, LP
   RET
```

SO, WITH THIS BACKGROUND, LISTING 3 SHOULD BE READILY COMPREHENSIBLE. THE TIMING IS NONE TO STARTLING, BUT IT IS A DEMO PROGRAM!

4. THE SAME THING IN ASSEMBLER:

THE ASSEMBLER CODE FOR THIS EXERCISE WOULD LOOK AS FOLLOWS -

```
LD A, 170D
LD HL, 7000H
LD DE, 7001H
LD BC, 7FFH
LD (HL), A
LDIR
RET
```

HOW WAS THIS PROGRAM AND LIST OF INSTRUCTIONS PUT TOGETHER? A PROCESS CALLED HAND ASSEMBLY WAS USED. (HAND ASSEMBLY IS USED FOR SHORT PROGRAMS - LONGER ONES ARE WRITTEN USING THE EDASM PROGRAM).

TO CARRY OUT HAND ASSEMBLY, THE FOLLOWING PIECES OF INFORMATION ARE REQUIRED -

-A NUMERIC LIST OF Z80 INSTRUCTIONS,
-AN ALPHABETIC LIST OF Z80 INSTRUCTIONS AND,
-A TABLE TO CONVERT DECIMAL (0 - 255) NUMBERS TO HEXADECIMAL (00H - 0FFH) NUMBERS.

SHOULD ANY USER REQUIRE SUITABLE COPIES OF THE TWO Z80 CODE LISTS, THEY CAN SEND ME $2-00 AND I WILL FORWARD COPIES BY RETURN MAIL. IT IS AN INTERESTING EXERCISE TO WRITE A BASIC PROGRAM THAT WILL PROVIDE A TABULATION OF DEC-HEX CONVERSION THAT IS SUITABLE FOR HAND ASSEMBLY.

LET'S EXAMINE HOW THIS ALGORITHM WORKS. YOU MAY WISH TO REFER TO LISTING 3 TO GAIN A COMPLETE UNDERSTANDING. THE BLOCK FILL IS REALLY BEST THOUGHT OF AS A "RIPPLE-FILL". THE SOURCE IS ALWAYS THE BYTE PRECEDING THE DESTINATION AND AS THE HL AND DE REGISTERS ARE INCREMENTED, A DESTRUCTIVE OVERWRITE OF THE PREVIOUS INFORMATION IN THE BLOCK OCCURS.

NOTE THAT THE VALUE, CONTAINED IN THE A REGISTER, MUST BE LOADED INTO THE FIRST MEMORY LOCATION OF THE FILL TO INITIALIZE THE PROCESS. THIS INITIALIZATION IS MOST OFTEN FORGOTTEN BY BEGINNERS. ANOTHER COMMON MISTAKE IS TO SET THE COUNT IN BC, ONE TOO HIGH BECAUSE THE FIRST BYTE IS INITIALIZED "OUTSIDE" OF THE LDIR INSTRUCTION. I TRUST THAT USERS CAN FOLLOW THE LOGIC OF THIS ALGORITHM - IF NOT, GO BACK TO LISTING 3 IN BASIC.

THE NEXT PROBLEM IN HAND ASSEMBLY IS TO FIND THE CORRESPONDING DECIMAL VALUES TO POKE INTO MEMORY AS MACHINE LANGUAGE. BY USING THE Z80 INSTRUCTION LISTS, MENTIONED PREVIOUSLY, A HEX EQUIVALENT OF THE ASSEMBLER CAN BE DERIVED. THESE ARE CONVERTED TO DECIMAL VALUES USING A CONVERSION TABLE (OR A CALCULATOR) AND ENTERED INTO DATA STATEMENTS. IT IS POSSIBLE TO PUT HEX VALUES INTO DATA STATEMENTS AND USE A HEX LOADER ROUTINE TO POKE THE VALUES INTO THE DESIGNATED AREA OF MEMORY.

THE ONLY REMAINING PROBLEM IS WHERE TO LOAD THE M/L? THIS HAS BEEN DISCUSSED IN A PREVIOUS ARTICLE ON FAST BASIC. FOR LISTING 4, I HAVE CHOSEN TO SIMPLY POKE IT INTO THE FREE SPACE LIST. (THIS IS A "LAZY" WAY, BUT I WILL FIX IT UP IN THE NEXT LISTING)! THE USR VECTOR IS SET TO THE START OF THE M/L.

SAVE THE PROGRAM TO DISK/TAPE BEFORE RUNNING THE BASIC PROGRAM. ASSEMBLER IS TOTALLY UNFORGIVING IF ANY ERRORS ARE MADE AND YOU RUN THE RISK OF HAVING TO RE-ENTER THE ENTIRE PROGRAM. (BE WARNED)!

LISTING 4 ACHIEVES THE HI-RES SCREEN FILL IN ABOUT 0.1 SECS. THIS IS WHAT WE WERE LOOKING FOR AND IT IS ACHIEVED WITH 15 BYTES OF M/L. I TRUST THAT THIS HAS GIVEN SOME NEW INSIGHT INTO ASSEMBLY LANGUAGE PROGRAMMING AND TAKEN SOME OF THE MYSTERY FROM IT. THIS ARTICLE ALSO PROVIDES A PAINLESS INTRODUCTION TO ASSEMBLY LANGUAGE PROGRAMMING.

5. MULTIPLE HI & LO RES SCREENS:

I HAVE PROVIDED A BONUS FOR THOSE WHO HAVE PERSEVERED THUS FAR! LISTING 5 IS THE LOGICAL CULMINATION OF THE EXERCISE WE SET ABOUT TO ACHIEVE AT THE START OF THIS ARTICLE. IT USES A SLIGHTLY MODIFIED FORM OF THE ROUTINE DESCRIBED ABOVE. IT PERMITS THE VALUE SELECTED TO FILL THE SCREEN TO BE PASSED, FROM THE BASIC PROGRAM, TO THE USR() ROUTINE. THE TECHNIQUE USED HAS BEEN DESCRIBED IN MY FAST BASIC ARTICLE. THE INTEGER VALUE CONTAINED IN THE BRACKETS OF THE USR STATEMENT IS PLACED INTO THE COMMUNICATION AREA AT ADDRESS 7921/2H AND CAN BE PICKED UP BY THE M/L PROGRAM.

THERE ARE THREE FURTHER FEATURES OF THIS PROGRAM THAT I HAVE NOT YET DISCUSSED AND ARE WORTHWHILE TO INCORPORATE INTO BASIC-M/L MODULES.

i. WHEN ENTERING M/L FROM DATA STATEMENTS, THE VALUES MUST BE EXACT. OTHERWISE, THE M/L ROUTINE IS WRONG AND A COMPUTER MALFUNCTION WILL OCCUR. (THE SAME THING HAPPENS IF THE USR POINTERS ARE NOT SET CORRECTLY). AS A CHECK THAT THE DATA STATEMENTS HAVE BEEN ENTERED CORRECTLY FROM A PROGRAM LISTING, IT IS USEFUL TO SET UP A "CHECKSUM" FACILITY. THIS SIMPLY KEEPS A RUNNING TOTAL OF THE DATA VALUES AS THEY ARE LOADED.

BEFORE EXECUTING THE PROGRAM, THE CHECKSUM IS COMPARED TO A CORRECT VALUE AND EXECUTION PREVENTED IF THE TWO VALUES DO NOT AGREE. IN LISTING 5, THE VARIABLE CS% IS USED IN LINES 210 TO 240. IT IS COMPARED WITH THE CORRECT VALUE IN LINE 250 AND IF NOT CORRECT PASSES TO AN ERROR HANDLING ROUTINE IN LINE 1000. IF THE ERROR MESSAGE IS SEEN, THEN THE DATA STATEMENTS IN LINES 410 TO 470 NEED TO BE CHECKED.

ii. HOW MANY PROGRAM LISTINGS HAVE YOU READ THAT SIMPLY LIST THE DATA VALUES TO BE POKED INTO MEMORY WITH NO EXPLANATION? ANY WONDER THAT BEGINNERS HAVE DIFFICULTIES IN UNDERSTANDING! THE METHOD OF SETTING OUT THE M/L IN "PSUEDO-ASSEMBLER" FORM IS TO BE STRONGLY RECOMMENDED.

iii. THE PROGRAM PLACES THE 16 BYTES OF M/L INTO A RESERVED AREA OF MEMORY CREATED BY LOWERING THE TOP-OF-MEMORY. IF AT ANY TIME, YOU FEEL AS IF THE M/L ROUTINE IS NOT FUNCTIONING PROPERLY, THEN IT IS A SIMPLE MATTER TO LOAD A DISASSEMBLER PROGRAM AND DECODE THE TOM AREA. THIS ENSURES THAT IT IS CORRECTLY LOADED, THAT YOUR DECIMAL VALUES CARRY OUT THE ACTION THAT YOU THINK THEY DO, AND THAT THE AREA IS NOT BEING OVERWRITTEN BY SOME OTHER PROCESSES. THIS IS A VERY POWERFUL FORM OF DEBUGGING WHEN DEVELOPING M/L ROUTINES.

LISTING 3:

```
010 '**************************************************
020 '***        VISIBLE Z-80 - DEMO OF LDIR          ***
030 '***   BOB KITCH - 7/3/91 - ASSEMBLER TUTOR      ***
040 '***        EXECUTION TIME: 70 SECONDS           ***
050 '**************************************************
060 '
070 'DEMONSTRATION OF SCREEN FILL IN BASIC THAT EMULATES
080 'THE LDIR INSTRUCTION OF THE Z80
```

```
090 'VARIABLES USED RESEMBLE THOSE OF THE Z80 REGISTER SET
100 '
140 MODE(1)                 :'***HI-RES SCREEN.
150 COLOR ,0                :'***GREEN BACKGROUND.
160 SOUND 10,1              :'***TIMING MARK.
170 '
180 '***ASSEMBLER SIMULATION STARTS HERE.
190 '***INITIALIZE ALL OF THE REGISTERS USED.
200 A%=170                  :'LD A,170    VALUE.
210 HL%=28672               :'LD HL,7000H SOURCE.
220 DE%=28673               :'LD DE,7001H DESTINATION.
230 BC%=2047                :'LD BC,07FFH COUNT.
280 '
290 '***PUT FIRST VALUE INTO START OF VIDEO RAM.
300 POKE HL%,A%             :'LD(HL),A
380 '
390 '***CARRY OUT DESTRUCTIVE BLOCK MOVE.
400 '                       :'LDIR
410 POKE DE%,PEEK(HL%)      :'    (DE) <- (HL)
420 HL%=HL%+1               :'     HL  <-  HL+1
430 DE%=DE%+1               :'     DE  <-  DE+1
440 BC%=BC%-1               :'     BC  <-  BC-1
450 IF BC%<>0 THEN GOTO 410:'      TEST FOR END
460 '                       :'RET
470 '
490 '***FINISH OFF.
500 SOUND 10,1              :'***TIMING MARK.
510 FOR I=0 TO 2000:NEXT I
520 STOP
600 END
```

LISTING 4:

```
001 '*************************************************************
002 '*   NEAR-LIGHT-SPEED GRAPHICS DEMO -- HI-RES VERSION 1.3   *
003 '*                  BY BOB KITCH - 22/5/86                  *
004 '*************************************************************
005 '
008 '*** EXECUTION TIME <0.5 SECS.
009 '***LOAD MACHINE CODE INTO FSL ABOVE BASIC VLT.
010 FOR I=-28687 TO -28673
020    READ A:POKE I,A
030 NEXT I
039 '
040 DATA 62,170       :'LD A,170     (#170D BLUE)
041 DATA 33,0,112     :'LD HL,7000H  (#28672D START VIDEO RAM)
042 DATA 17,1,112     :'LD DE,7001H  (#28673D NEXT)
043 DATA 1,255,7      :'LD BC,07FFH  (#2047D  SIZE OF VIDEO RAM)
044 DATA 119          :'LD (HL),A
045 DATA 237,176      :'LDIR         (BLOCK LOAD COMMAND)
046 DATA 201          :'RET
047 '
049 '***INITIALIZE USR() TO ADDRESS 8FF1H OR #-28687D IN FSL.
050 POKE 30862,241:POKE 30863,143
058 '
059 '***PUT UP BLUE SCREEN.
060 MODE(1):COLOR,0
070 SOUND 10,1
080 X=USR(0)
090 SOUND 10,1
098 '
099 '***DELAY TO VIEW SCREEN.
100 FOR I=0 TO 2000:NEXT I
110 END
```

INFORMATION ABOUT QUICKWRITE
BY LESLIE MILBURN

TO MY SUPRISE IN THE LAST COUPLE OF WEEKS I HAVE RECEIVED A FEW ENQUIRIES ABOUT QUICKWRITE. BELOW IS THE ANSWERS TO MOST OF THE QUESTIONS WHICH I FEEL MAY INTEREST OTHER PEOPLE.

WHAT DO THE FILENAMES MEAN?

THE RELEASE NUMBERING I USED IS AS FOLLOWS:-

QW A.B.C

WHERE:-

QW - THIS IS THE APPLICATION ID.
A - THIS IS THE VERSION OF THE APPLICATION.
B - THIS INDICATES THE PROGRAM STRUCTURE LEVEL.
(I.E. IMPROVEMENTS MADE TO EXISTING FACILITIES).
C - THIS INDICATES THE NUMBER OF BUG CORRECTIONS APPLIED.

PROGRAMMERS SHOULD NOTE THAT FILENAMES SHOULD BE CHOSEN WITH CARE.

WHAT HAS BEEN RELEASED?

THERE WERE 3 MAJOR VERSIONS OF QUICKWRITE RELEASED, THESE WERE:-

QW3 - A LOW MEMORY DISK BASED WORD PROCESSOR;
QW4 - AN ENHANCED VERSION OF QW3 WITH IMPROVED DISK & PRINTING FACILITIES; AND LASTLY
QWII - A GENERAL PURPOSE EDITOR WHICH MADE USE OF BANK SWITCHED MEMORY.

THERE WERE 6 FREE UPGRADES OF QW3 AND 3 FREE UPGRADES OF QW4. ONLY 1 VERSION OF QWII WAS EVER RELEASED. THE LAST RELEASED VERSIONS OF THESE WERE:-

QW3.3.3; QW4.2.2 & QWII.4.7.

WHY DOES QUICKWRITE II TEXT FILES HAVE A START ADDRESS OF 0000H & END ADDRESS OF FFFFH?

THE ANSWER IS THAT THE FILE FORMAT OF QWII FILES CHANGED EVEN THOUGH THE FILETYPE IS THE SAME. ALL QW TEXT FILES HAVE FILETYPE "F".

QW3 & QW4 BOTH USE THE DKLOAD DOS FUNCTION TO LOAD A DOCUMENT. THE START ADDRESS IN THE DIRECTORY IS ACTUALLY 10 BYTES LESS THAN THE START OF THE ACTUAL TEXT. I.E. EACH FILE HAS A 10 BYTE HEADER. ONLY 4 BYTES OF THIS HEADER WERE EVER USED. THESE ARE:-

HEADER	SIZE (BYTES)	DESCRIPTION
2	2	THE LENGTH OF THE DOCUMENT.
6	2	THE CURSOR POSITION (ONLY USED BY EARLIER VERSIONS).

QUICKWRITE II HAS ITS OWN DISK ROUTINES WHICH TAKE BANK SWITCHED MEMORY INTO ACCOUNT. IT DOES NOT USE THE START AND END ADDRESS IN THE DIRECTORY. TO LOAD A FILE IT LOADS ALL SECTORS BY FOLLOWING THE TRACK/SECTOR POINTERS. THE START ADDRESS OF 0 & END ADDRESS OF FFFFH ARE STORED IN THE DIRECTORY PURELY TO PREVENT QW3 & QW4 FROM TRYING TO LOAD THE FILE. THE FIRST SECTOR OF EACH QWII FILE IS USED TO STORE INFORMATION ABOUT THE FILE. THE CONTENTS OF THIS SECTOR ARE AS FOLLOWS:-

OFFSET	SIZE (BYTES)	DESCRIPTION
0	2	THE TOTAL LENGTH OF THE FILE.
2	2	THE NO. OF BYTES IN THE LAST BANK USED.
4	1	THE DEFAULT FILE FORMAT (1=LEFT JUST.)
5	1	THE LEFT MARGIN WIDTH.
6	2	BOTH BYTES ZERO. THIS INDICATES THAT THE FILE IS A QWII FILE.
8	1	PAGE NUMBERING FLAG. (1=YES).
9	1	MAX. NO. OF PRINTER COLUMNS.
10	2	LINE LENGTH OF OUTPUT.
12	2	SIZE OF MARGIN AT TOP OF PAGE.
14	2	NO. OF LINES PER PAGE.
16	2	SIZE OF MARGIN AT BOTTOM OF PAGE.
18	2	START PAGE NUMBER.
20	2	INTERNAL BUFFER POINTER.
22	2	INTERNAL BUFFER POINTER.
24	1	WIDTH OF A TAB MARKER.
25	1	INDENT/TAB SELECTION FLAG.

THE REMAINDER OF THE SECTOR IS CURRENTLY UNUSED.

CAN QUICKWRITE BE MODIFIED TO WORK WITH MORE THAN 3 BANKS OF MEMORY?

IN THEORY QWII COULD WORK WITH ANY NUMBER OF BANKS. HOWEVER, WHEN WRITING THE SYSTEM I MADE A MISTAKE BY HARD CODING THE MAXIMUM NUMBER OF BANKS IN THE CODE. IT IS NOT JUST A CASE OF CHANGING THE NUMBER AND REASSEMBLING. ALL FUNCTIONS WHICH USE THE CLIPBOARD WOULD HAVE TO BE MODIFIED (I.E. CUT, COPY, PASTE, RUBOUT, INSERT). AT THIS STAGE I CANNOT SEE MYSELF DOING THIS. ALSO NOT MANY PEOPLE HAVE 64K RAMS LET ALONE MORE MEMORY.

WILL THERE BE ANY MORE RELEASES OF QUICKWRITE?

NO, NOT IN THE FORSEEABLE FUTURE. HOWEVER I HOPE TO DO A SERIES OF ARTICLES WHICH WILL USE QW CODE AS THEIR BASIS. THE IDEA IS TO MAKE THE CODE APPLICATION INDEPENDANT. THE FIRST OF THESE ARTICLES WAS KSCAN WHICH WAS IN THE LAST JOURNAL.

WHERE CAN I GET A COPY OF QUICKWRITE?

ALL VERSIONS OF QW CAN NOW BE FREELY COPIED AMONG YOURSELVES. THE DISKS WERE PROTECTED USING DISK GUARD. IF YOU CANNOT GET HOLD OF AN UNPROTECTED COPY GET IN CONTACT WITH:-

J.P. LEON 33 TIGHES TERRACE TIGHES HILL 2297 (049) 692 399

CAN QUICKWRITE TEXT FILES BE USED BY A DISK BASED E&F WORD PRO?

A CONVERSION PROGRAM WOULD HAVE TO BE WRITTEN. WITH THE INFORMATION GIVEN ABOVE THE FILE CAN BE EASILY READ. ALL THAT IS REQUIRED IS TO CONVERT ANY SPECIAL CHARACTERS. THE SPECIAL CHARACTERS ARE AS FOLLOWS:-

NEW PAGE MARKER	= 222
TAB/INDENT MARKER	= 254
CARRIAGE RETURN MARKER	= 252
FORMAT CODES	= 241-244
USER DEFINED CHARS	= 225-233

* * * DAVE MITCHELL SOFTWARE * * *

E & F WP PATCH 3.3: PATCH 3.3 WRITTEN BY DAVE MITCHELL WILL CONVERT YOUR E & F TAPE WORD PROCESSOR FOR FULL DISK USE WHILE RETAINING ALL ORIGINAL FUNCTIONS. IT ALSO HAS SHIFT LOCK AND PRINTER CONTROL CODES WHICH CAN BE IMBEDDED IN TEXT AND SAVED TO TAPE OR DISK.

BSTWP.F: THIS UTILITY PROVIDED WITH PATCH 3.3 WILL CONVERT BASIC PROGRAMS AND ED/ASS. SOURCE CODE FILES INTO WORD PROCESSOR FILES.

PRICE: AUS/NZ AU$20.00 - UPDATE - AUS-$10.00 - NZ-AU$11.00.

EXTENDED DOS V1.3: THESE COMMANDS ARE AT YOUR DISPOSAL: MERGE, DIRA, DIRA, DIRB, LDIRB, OLD, OLD., DEC, HEX, MENU, CODE, LTAB, MOVE AND UPDATE, STATUSA AND LSTATUSA. STATUSA AND LSTATUSA ALSO WORKS WITH VERSION 1.0 DOS

PRICE: AU$15.00 - POSTAGE INCLUDED

MENU/FILE COPIER: THIS UTILITY WILL READ YOUR DISK DIRECTORY AND PRESENT YOU WITH SEVERAL OPTIONS. USING THE CURSOR YOU CAN RUN/BRUN ANY PROGRAM OR SELECT FILE COPY, REN, ERASE, DRIVE 1 OR 2, ETC. BESIDES COPYING TEXT AND BINARY FILES ALL OTHER FILES CAN BE COPIED AS WELL EXEPT FOR DATA FILES.

PRICE: AU$15.00 - POSTAGE INCLUDED

FOR PURCHASE OR INFORMATION CONTACT:

DAVE MITCHELL 24 ELPHINSTONE STREET
NORTH ROCKHAMPTON QUEENSLAND 4701
AUSTRALIA - PHONE: (079) 27 8519

* * * PETER HICKMAN SOFTWARE * * *

VZ DISASSEMBLER: WHAT, ANOTHER DISASSEMBLER? BUT, YOU HAVE ALREADY GOT ONE? THIS ONE IS DIFFERENT! THIS PROGRAM IS ENTIRELY WRITTEN IN MACHINE CODE. IT ACTUALLY RUNS ABOUT 40 TIMES FASTER THAN D.S.E.'S DISASSEMBLER (OR ANY ONE ELSE'S). IT WILL DISASSEMBLE ANY PROGRAM THAT YOU CAN BLOAD INTO MEMORY. IT WORKS WITH ANY VZ CONFIGURATION. IT DISASSEMBLES EVEN THE 88 EXTRA Z80 OPCODES THAT ZILOG DOESN'T ADMIT TO.

PRICE: AU$25.00 - PRICE INCLUDES HARDCOPY MANUAL.
TAPE AND DISK VERIONS AVAILABLE.

VZ MODEM SOFWARE: DID YOU WANT TO TALK TO OTHER COMPUTERS VIA A MODEM? DID YOU BUY THE DSE TERMINAL EPROM, ONLY TO DISCOVER THAT IT ONLY WORKS WITH TAPE. IT ONLY ALLOWS YOU TO PRINT FILES, NOT SAVE THEM OR SEND THEM!

YOUR PROBLEMS ARE SOLVED! THE HICKMAN BROTHERS, PETER AND ANDREW, HAVE A BRAND NEW PROJECT WHICH WILL ALLOW YOU TO SEND, RECEIVE & SAVE FILES VIA A MODEM. IT WORKS WITH DISK!

SALE PRICE: $25.00 - INCLUDED ARE INSTRUCTIONS FOR THE HARDWARE MODIFICATIONS. A SMALL MODIFICATION IS NEEDED TO YOUR DISK CONTROLLER. YOUR USER GROUP MAY HELP YOU MODIFY YOUR COMPUTER TO USE THIS EXCITING NEW SOFTWARE!

THE MANUAL IS SUPPLIED ON DISK FOR PRINTING OUT WITH YOUR DISK VERSION OF E & F W/PROCESSOR. IF YOU DO NOT OWN AN E & F W/PROCESSOR THEN PLEASE ENCLOSE ANOTHER $5.00 (TOTAL $30.00) FOR PHOTOCOPYING AND POSTAGE OF THE MANUAL.

FOR PURCHASE OR INFORMATION CONTACT:
PETER HICKMAN PO BOX 8 WERRINGTON 2747

* * CONTRIBUTIONS TO THE JOURNAL * *

IF YOU ARE THINKING OF CONTRIBUTING TO THE JOURNAL THE PREFERED FORMAT IS BASIC LISTINGS, WORD PROCESSOR OR SOURCE CODE FILES ON TAPE OR DISK. FILES FROM THE FOLLOWING WORD PROCESSORS CAN BE ACCEPTED :-

E & F TAPE OR DISK PATCH 3.1-3.3, WORDPRO CARTRIDGE, WORDPRO PATCH, MOST SOURCE CODE FILES AND ALL QUICKWRITE WORD PROCESSOR FILES.

* * CLUB MEETINGS - ALL WELCOME * *

FIRST FRIDAY OF MONTH

* * FUTURE MEETINGS - NEW VENUE * *

AS MENTIONED BEFORE WE NO LONGER MEET AT JNC, BUT AT VARIOUS MEMBERS HOMES. MEETINGS WILL BE ONCE A MONTH AS BEFORE WITH THE DATES BEING FIRST FRIDAY OF THE MONTH.

BECAUSE OF SOME LOCAL MEMBERS HAVING TO WORK SHIFTWORK MEETING DATES WILL BE ADJUSTED TO ACCOMODATE THEM. WHETHER YOU ARE A LOCAL MEMBER, INTRA OR INTERSTATE VISITOR PLEASE CHECK WITH JOE LEON FIRST BEFORE COMING OUT.

JOE LEON 33 TIGHES TCE TIGHES HILL 2297 (049) 692 399

* CLUB COMMITTEE & SUBSCRIPTIONS *

PRESIDENT - ROSS WOODS - SECRETARY/EDITOR - JOE LEON
COMMITTEE MEMBERS - COLIN BRIDGE - PETER JONES

SUBSCRIPTION TO - AUST. - 3 ISSUES $11.00 - 6 ISSUES $21.00
H.V.VZ.JOURNAL - N. Z. - 3 ISSUES $13.00 - 6 ISSUES $26.00

FOR MORE INFORMATION CONTACT:

JOE LEON 33 TIGHES TCE TIGHES HILL 2297 (049) 692 399 AUSTRALIA

NOTE: PRICES INCLUDE POST & PACKING

* * VZ USER GROUPS & PUBLICATIONS * *

VZ DOWN UNDER - VZ MAGAZINE - 6 ISSUES - $18.00 PER ANUM
HARRY HUGGINS 12 THOMAS SREET MITCHAM VICTORIA 3132

WAVZ - WESTERN AUSTRALIA VZ USER GROUP
GRAEME BYWATER PO BOX 388 MORLEY W A 6062

BRISBANE VZ USERS WORKSHOP - C/O 63 TINGALPA ST. WYNUM WEST 4178
SOFTWARE FOR SALE - DISK MENU

SAPPHIRE PRODUCTIONS - VZ DISK MAGAZINE - PUBLIC DOMAIN
NOTE: VZ DISK MAGAZINE HAS CEASED PRODUCTION

NOTE: WHEN WRITING TO ANY ABOVE OR H.V.VZ. USERS' GROUP FOR INFORMATION PLEASE ENCLOSE A S.S.A.E. OR NZ 2 INT. REPLY COUPONS.